Characteristics-based Simulink Implementation of First-order Quasilinear Partial Differential Equations

Anton Ponomarev[1][a], Julian Hofmann[2][b] and Lutz Gröll[2][c]

[1]*Saint Petersburg State University, 7/9 Universitetskaya nab., St. Petersburg, 199034, Russia*

[2]*Karlsruhe Institute of Technology, Institute for Automation and Applied Informatics, Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany*

Keywords: Partial Differential Equations, Simulink, Transport, Convection, Method of Characteristics.

Abstract: The paper deals with solving first-order quasilinear partial differential equations in an online simulation environment, such as Simulink, utilizing the well-known and well-recommended method of characteristics. Compared to the commonly applied space discretization methods on static grids, the characteristics-based approach provides better numerical stability. Simulink subsystem implementing the method of characteristics is developed. It employs Simulink's built-in solver and its zero-crossing detection algorithm to perform simultaneous integration of a pool of characteristics as well as to create new characteristics dynamically and discard the old ones. Numerical accuracy of the solution thus obtained is established. The subsystem has been tested on a full-state feedback example and produced better results than the space discretization-based "method of lines". The implementation is available for download and can be used in a wide range of models.

1 INTRODUCTION

Dynamical systems described by first-order quasilinear partial differential equations (PDEs) are commonly interpreted as convective-reactive processes, i.e., they represent transport phenomena with sources and sinks. Such phenomena play an important role in chemical and process engineering, traffic flow models, material science, biology, etc. (Mahmoudi et al., 2019; Kachroo and Özbay, 2018; Ingham and Pop, 1998; Truskey et al., 2004). PDEs of this form typically arise as mathematical formulation of the physical conservation laws of spatially distributed quantities (mass, momentum, etc.) applied locally (to an infinitesimal volume) in which case they are also known as continuity equations (Greenkorn, 2018).

In this paper we discuss computer simulation of arbitrary convection-reaction dynamics described by the following first-order quasilinear PDE on a bounded spatial domain $[0,\ell]$:

$$w_t + v\big(t,x,w(t,\cdot)_{[0,\ell]}\big)w_x = f\big(t,x,w(t,\cdot)_{[0,\ell]}\big), \tag{1a}$$

$$w(0,x) = w_0(x),\ x\in[0,\ell], \tag{1b}$$

$$w(t,0) = u(t) \tag{1c}$$

where $t\geq 0$, $x\geq 0$, $\ell\in(0,\infty)$, $w(t,x)$ is the internal state of the system, w_t and w_x are partial derivatives of w at the point (t,x), w_0 is an L^2 (square-integrable) initial state function, v represents the convection velocity ($v\geq\varepsilon>0$ for all possible arguments), f can be interpreted as a source, sink, or reaction term, and u plays the role of a boundary input into the system. Symbol $w(t,\cdot)_{[0,\ell]}$ stands for the restriction of the function $w(t,x)$ to $x\in[0,\ell]$.

The terms v and f in (1) depend on time t, spatial position x, and full system state $w(t,\cdot)_{[0,\ell]}$. In addition, they may also depend on other signals, e.g., on the output of a dynamic feedback controller. The same holds true for the boundary input $u(t)$.

Remark 1. From the theoretical standpoint, it would be enough to define functions v and f only for $x\in[0,\ell]$. However, our simulation algorithm described below requires that the system be defined on a larger domain $x\in[0,\bar\ell]$ with some $\bar\ell>\ell$ (see Remark 6). Although a sufficiently large but finite $\bar\ell$ could be estimated, for the sake of brevity we opt to assume that the system is defined for all $x\geq 0$. In the practical sense, it means that the functions v and f should be extended from $x\in[0,\ell]$ to $x>\ell$ in a physically meaningful way.

Remark 2. The system (1) admits unique classical solution (i.e., well-defined single-valued function)

[a] https://orcid.org/0000-0002-8753-9759
[b] https://orcid.org/0000-0003-3210-4368
[c] https://orcid.org/0000-0003-3433-1425

Ponomarev, A., Hofmann, J. and GrÃ¶ll, L.
Characteristics-based Simulink Implementation of First-order Quasilinear Partial Differential Equations.
DOI: 10.5220/0009569001390146
In *Proceedings of the 10th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2020)*, pages 139-146
ISBN: 978-989-758-444-2

if there are no intersecting characteristics (Polyanin et al., 2002). Otherwise, the so-called shock waves appear which can be dealt with using the concept of multi-valued solutions or by introducing discontinuity at the shock wave's front. In this paper we avoid making restrictive a priori assumptions that would exclude the possibility of intersecting characteristics. Furthermore, our algorithm will not handle the shock waves by itself. Instead, the algorithm will detect intersecting characteristics during runtime and inform the user of the problem.

Let us discuss the existing approaches to computer simulation of the dynamical system (1) in the MATLAB/Simulink environment paying primary attention to the method of characteristics (Polyanin et al., 2002) as it is arguably the most accurate and efficient tool to solve a general PDE like (1a).

There are numerous MATLAB scripts that implement the method of characteristics, e.g., (von Petersdorff, 2012). Assuming that the functions v, f, and u are given as functions of t, x, and (to some extent) $w(t,\cdot)_{[0,\ell]}$, such MATLAB scripts can be used to calculate solutions of the systems of the form (1). However, these solvers are "offline" in the sense that the state $w(t,x)$ is available for all values of t and x only after the script is executed completely. We cannot, in a straightforward manner, embed such an "offline" solver into a Simulink model as a *MATLAB Function* block because it will not output the full state $w(t,\cdot)_{[0,\ell]}$ as the simulation time t increases.

We are interested in an "online" solver, which is to say that, as the simulation time t increases, it should continuously produce an approximation of the current state $w(t,\cdot)_{[0,\ell]}$. An "online" solver is required if at each moment in time the terms v, f, and u in (1) depend on the full system state $w(t,\cdot)_{[0,\ell]}$ in a non-trivial manner or are defined via feedback of the full system state through another dynamical system which may be the case, e.g., under feedback control.

Simulink is a suitable environment to implement (1) in the "online" fashion. Having a block which takes the values of v, f, and u and outputs a discrete approximation of the full system state $w(t,\cdot)_{[0,\ell]}$, it would be easy to plug it into a larger system. However, we are not aware of Simulink implementations of the method of characteristics for the general case of system (1). Let us describe some alternatives that are available in Simulink at the moment and motivate our effort to implement the method of characteristics.

Space discretization (finite difference) method, also known as the *Method of Lines* (Schiesser, 2012), can be used to transform the PDE into a system of ordinary differential equations (ODEs) which has native Simulink implementation. In general, these approaches suffer from instability because of numerical diffusion and dispersion (Zijlema, 2015). We illustrate this problem in Section 4.

Another approach is to reformulate the PDE as a time-delay system. For example, (Witrant and Niculescu, 2010) consider a variable-velocity transport system with sinks or sources

$$\begin{aligned} w_t + v(t)w_x &= f(t), \\ w(t,0) &= u(t), \\ y(t) &= w(t,1). \end{aligned} \tag{2}$$

Using the method of characteristics, it is transformed into a delay-differential equation which can be plugged into a Simulink model. The special case of (2) with $f(t)=0$ is included in Simulink under the name of *Variable Transport Delay* block (Zhang and Yeddanapudi, 2012). In general, however, converting a model like (1) into a time-delay form requires deep analysis and may not always be possible.

Finally, there are examples of connecting Simulink to dedicated PDE solving environments (Van Schijndel, 2009) but such an approach is only justified for simulating complex phenomena. For a simple transport process the overhead is excessive.

Hence, we set the goal of designing a simple but versatile Simulink block which can simulate the dynamics of a wide range of first-order quasilinear PDEs based on the method of characteristics. The block is to be employed in modeling the systems like (1). The accuracy of simulation should be adjustable via some parameters. The block has to be easy to use without extensive analysis of the mathematical model. Furthermore, it must have straightforward interface and be easily embeddable into larger Simulink models.

Remark 3. To the best of the authors' knowledge, the only example that comes close to reaching the goal of this paper is (Herrán-González et al., 2009). It is a library of Simulink blocks modeling gas ducts, valves, compressors, etc. Although the blocks indeed implement the method of characteristics, the library is directed exclusively at modeling gas distribution pipeline networks which results in a limited choice of possible dynamics. Our approach is more general as we model dynamics (1) in an abstract sense. We allow arbitrary velocity and source terms as well as initial functions.

The plan of the paper is as follows. In Sec. 2 we provide theoretical background to the proposed simulation approach. Specifically, in Sec. 2.1 we recall the method of characteristics whereas in Sec. 2.2 we introduce its algorithmic realization suitable for "online" simulation and assess the accuracy of the algorithm. Sec. 3 describes implementation of the algorithm as a Simulink block. In Sec. 4, performance of the block is compared with the method of lines using an illustrative example.

2 METHOD

2.1 The Method of Characteristics

Our approach to simulation of the dynamics (1) is based on the method of characteristics (Polyanin et al., 2002). Let us recall the definition of the characteristics.

Definition. The *characteristics* of the system (1) starting at the point (t_0,x_0) on the initial boundary ($t_0 = 0$ and $x_0 \geq 0$) or on the input boundary ($t_0 > 0$ and $x_0 = 0$) are the solutions $\xi(t;t_0,x_0)$ and $\omega(t;t_0,x_0)$ of the system of ordinary differential equations (ODEs)

$$\begin{aligned}\dot{\xi}(t;t_0,x_0) &= v\big(t,\xi(t;t_0,x_0),w(t,\cdot)_{[0,\ell]}\big),\\ \dot{\omega}(t;t_0,x_0) &= f\big(t,\xi(t;t_0,x_0),w(t,\cdot)_{[0,\ell]}\big)\end{aligned} \tag{3}$$

where $t \geq 0$, with initial conditions

$$\xi(0;0,x_0) = x_0, \quad \omega(0;0,x_0) = w_0(x_0), \quad x_0 \geq 0$$

(in the case of initial boundary) or

$$\xi(t_0;t_0,0) = 0, \quad \omega(t_0;t_0,0) = u(t_0), \quad t_0 > 0$$

(in the case of input boundary). Here $\dot{\xi}$ and $\dot{\omega}$ denote the derivatives of ξ and ω with respect to t.

The functions ξ and ω being a solution of (3), we know from (Polyanin et al., 2002) that the solution $w(t,x)$ of (1) takes the values $\omega(t;t_0,x_0)$ along the curve $\big(t,\xi(t;t_0,x_0)\big)$, i.e.,

$$w\big(t,\xi(t;t_0,x_0)\big) = \omega(t;t_0,x_0). \tag{4}$$

Thus, starting at a number of points (t_0,x_0) on the initial and input boundaries and integrating the characteristic ODEs (3) one can find the solution of the PDE problem (1) along a number of curves spanning the domain. However, the structure of (1) is such that the full state $w(t,\cdot)_{[0,\ell]}$ of the PDE is involved in the characteristic ODEs (3). Therefore, the ODEs (3) pertaining to different characteristics are to be integrated together (in parallel, so to speak) while the function $w(t,\cdot)_{[0,\ell]}$ is interpolated using the relations (4). In the next Section we clarify this approach.

2.2 Simulation Algorithm

Our proposal is to start simulation at $t = 0$ with a set of characteristics spread across the spatial domain and integrate their respective ODEs (3) in parallel. To this end, the full state $w(t,\cdot)_{[0,\ell]}$ must be plugged into the right-hand side of (3). However, the full state is unknown. Only its values on the characteristic curves are available, according to (4). Therefore, we will approximate $w(t,\cdot)_{[0,\ell]}$ in (3) via interpolation between the characteristics. Once a characteristic has moved far enough out of the spatial domain and has no effect on the full PDE state anymore, it is removed from the set. New characteristics are created on the input boundary and added to the set when appropriate according to some rules given below.

To give a precise description of the algorithm, let us represent the set of characteristics as two variable-length vectors $\Xi(t)$ and $\Omega(t)$ each of which at time t contains $N(t) \geq 2$ elements:

$$\begin{aligned}\Xi(t) &= \big(\xi_1(t),\xi_2(t),\ldots,\xi_{N(t)}(t)\big),\\ \Omega(t) &= \big(\omega_1(t),\omega_2(t),\ldots,\omega_{N(t)}(t)\big).\end{aligned} \tag{5}$$

These vectors are to be handled by the following hybrid algorithm which combines continuous integration with event-triggered state resets.

Algorithm. Given parameters $\Delta_x \in (0,\ell)$, $\Delta_w > 0$, and $\Delta_t > 0$, the rules for the simulation of (1) are:

1. *Initialization*: initial number of characteristics $N(0) \geq 2$ and the elements of the vectors $\Xi(0)$ and $\Omega(0)$ are selected to approximate the initial function such that
$$\begin{aligned}0 = \xi_1(0) < \xi_2(0) < \cdots < \xi_{N(0)}(0) = \ell,\\ \omega_i(0) = w_0\big(\xi_i(0)\big) \quad \big(i = 1,2,\ldots,N(0)\big)\end{aligned} \tag{6}$$
and
$$\begin{gathered}\xi_{i+1}(0) - \xi_i(0) \leq \Delta_x,\\ |w_0(x) - \omega_i(0)| \leq \Delta_w\\ \big(x \in [\xi_i(0),\xi_{i+1}(0)], \quad i = 1,2,\ldots,N(0)-1\big).\end{gathered}$$
Moreover, an auxiliary variable $t_{\mathrm{LC}}(t)$, denoting the last time when new characteristic was created, is initialized as $t_{\mathrm{LC}}(0) = 0$.
2. *Dynamics* (continuous integration): $\Xi(t)$ and $\Omega(t)$ evolve according to the equations similar to (3):
$$\begin{gathered}\dot{\xi}_i(t) = v\big(t,\xi_i(t),\tilde{w}(t,\cdot)_{[0,\ell]}\big),\\ \dot{\omega}_i(t) = f\big(t,\xi_i(t),\tilde{w}(t,\cdot)_{[0,\ell]}\big)\\ \big(i = 1,2,\ldots,N(t)\big)\end{gathered} \tag{7}$$
with initial conditions (6). The ODEs (7) are obtained from (3) by substitution, in place of the unknown $w(t,\cdot)_{[0,\ell]}$, its approximation $\tilde{w}(t,\cdot)_{[0,\ell]}$. The latter function is an interpolant over the grid consisting of a point on the input boundary and $N(t)$ points on the characteristic curves, as suggested by (4):
$$\begin{aligned}\tilde{w}(t,0) &= u(t),\\ \tilde{w}\big(t,\xi_i(t)\big) &= \omega_i(t) \quad \big(i = 1,2,\ldots,N(t)\big).\end{aligned} \tag{8}$$

Depending on the expected analytic properties of the solution, a suitable interpolation scheme should be chosen here: nearest neighbor, linear, spline, etc.
The values $N(t)$ and $t_{\mathrm{LC}}(t)$ are kept constant during integration of the ODEs.

3. *Removal trigger* (state reset): fulfillment of the condition
$$\xi_{N(t)-1}(t) \geq \ell \tag{9}$$
triggers removal of the oldest ($N(t)$'th) characteristic from the set:
$$N(t+0) := N(t) - 1 \tag{10}$$
where the argument $(t+0)$ indicates the updated value of the variable during the state reset.
4. *Creation triggers* (state reset): fulfillment of any of the conditions
$$\xi_1(t) \geq \Delta_x, \tag{11a}$$
$$|\omega_1(t) - u(t)| \geq \Delta_w, \tag{11b}$$
$$t - t_{\mathrm{LC}}(t) \geq \Delta_t \tag{11c}$$
triggers creation of a new characteristic at the input boundary and resetting t_{LC}:
$$\begin{aligned} N(t+0) &:= N(t)+1, \\ \xi_1(t+0) &:= 0, \\ \xi_{i+1}(t+0) &:= \xi_i(t), \\ \omega_1(t+0) &:= u(t), \\ \omega_{i+1}(t+0) &:= \omega_i(t), \\ t_{\mathrm{LC}}(t+0) &:= t \\ &\big(i = 1,2,\ldots,N(t)\big). \end{aligned}$$
5. *Output*: the algorithm outputs vectors $\Xi(t)$ and $\Omega(t)$ which can be used to construct an approximation $\tilde{w}(t,\cdot)_{[0,\ell]}$ of the full PDE state $w(t,\cdot)_{[0,\ell]}$ by interpolation over the grid (8).

Remark 4. Assuming exact trigger detection, the algorithm preserves the condition $N(t) \geq 2$ because characteristic creation trigger (11a), thanks to the requirement $\Delta_x < \ell$, ensures that $\xi_1(t) < \ell$ for all t which guarantees that removal condition (9) cannot be satisfied if $N(t) = 2$. Thus, $N(t)$ cannot drop below 2.

Straightforward estimations of the solutions of the characteristic equations (3) yield the following statements regarding the accuracy of the Algorithm.

Theorem. *Suppose the PDE (1a) has the form*
$$w_t + v\big(t,x,w(t,x)\big)w_x = f\big(t,x,w(t,x)\big) \tag{12}$$
and there exist positive constants $\hat{T}$ and $\hat{F}$ such that the inequalities
$$v(t,x,w) \geq \frac{\ell}{\hat{T}},$$
$$\left\| \begin{matrix} v(t,\tilde{x},\tilde{w}) - v(t,x,w) \\ f(t,\tilde{x},\tilde{w}) - f(t,x,w) \end{matrix} \right\| \leq \hat{F} \left\| \begin{matrix} \tilde{x} - x \\ \tilde{w} - w \end{matrix} \right\|$$
hold for all values of $t, x, \tilde{x} \geq 0$ and $w, \tilde{w}$ where $\|\cdot\|$ denotes Euclidean vector norm in $\mathbb{R}^2$. Then, assuming that ODEs (7) are integrated exactly and triggers (9), (11) are detected perfectly on time, the following estimation is valid for all $t \geq 0$ and $x \in [\xi_i(t), \xi_{i+1}(t)]$, $i = 1,2,\ldots,N(t)-1$:
$$\left\| \begin{matrix} x - \xi_{i+1}(t) \\ w(t,x) - \omega_{i+1}(t) \end{matrix} \right\| \leq \mathrm{e}^{(\hat{T}+\Delta_t)\hat{F}} \left\| \begin{matrix} \Delta_x \\ \Delta_w \end{matrix} \right\|$$
where $w(t,x)$ is the exact solution of the problem (1) and functions $\xi_i(t)$ and $\omega_i(t)$ are the outputs of the Algorithm.

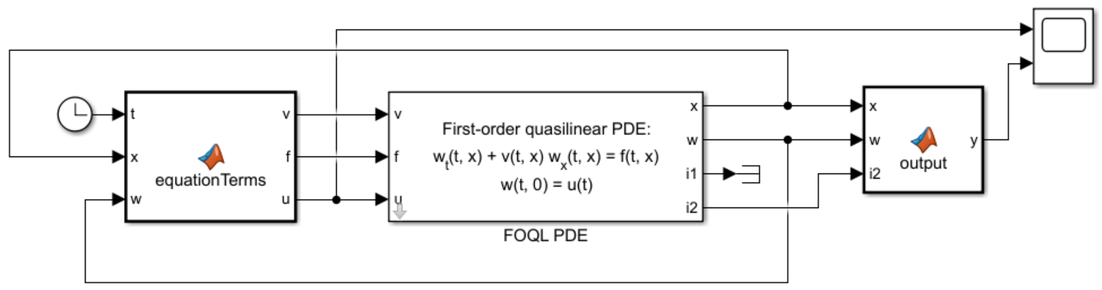
Corollary. *Under the assumptions of the Theorem, the following holds for all $t \geq 0$ and $x \in [0,\ell]$:*
$$\big|w(t,x) - \tilde{w}(t,x)\big| \leq 2\,\mathrm{e}^{(\hat{T}+\Delta_t)\hat{F}} \left\| \begin{matrix} \Delta_x \\ \Delta_w \end{matrix} \right\|$$
where $w(t,x)$ is the exact solution of the problem (1) and $\tilde{w}(t,x)$ is the approximation of $w(t,x)$ obtained via linear or nearest-neighbor interpolation over the grid (8).

Remark 5. The Theorem assumes that the PDE (1a) has the form (12) which excludes dependence of the dynamics on the full state of the system. Such full-state feedback may, in general, lead to divergence of the Algorithm and there would be no time-invariant accuracy estimation (although it can be established when $\hat{T}\hat{F}$ is small enough). This is due to an interpolant being used in place of the full PDE state in the characteristic ODEs (7) which leads to unbounded error accumulation. Nonetheless, as can be seen in the example of Section 4, the Algorithm may produce good results even in presence of full-state feedback.

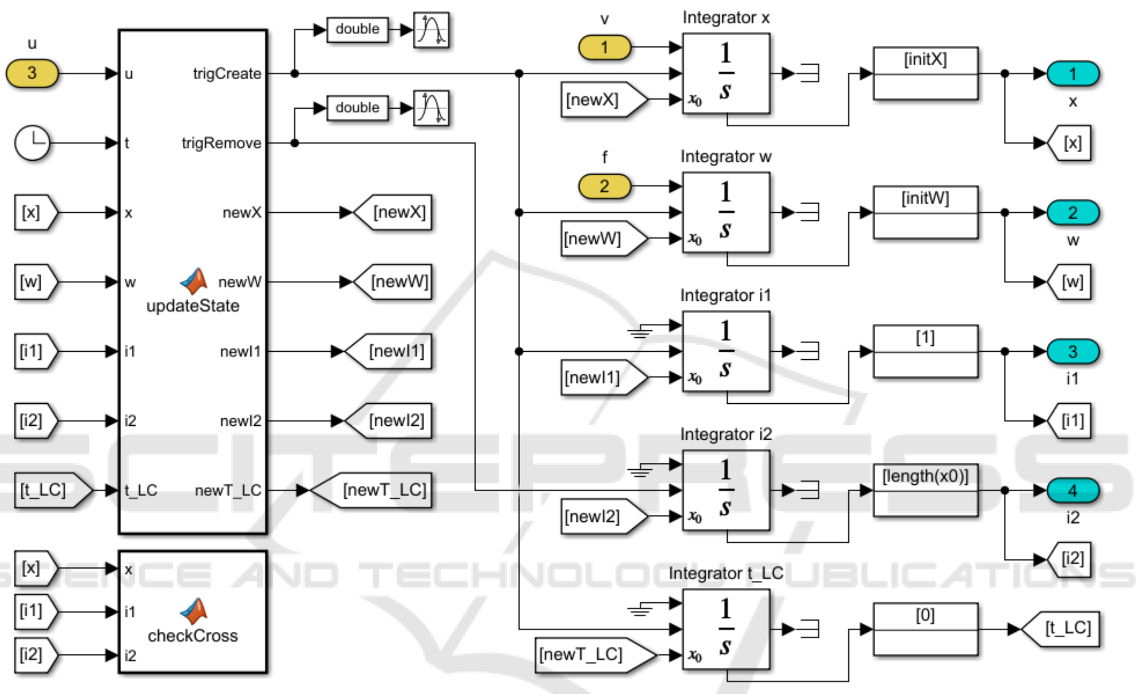
Remark 6. The Algorithm implies that the oldest ($N(t)$'th) characteristic lies outside the spatial domain $[0,\ell]$, i.e., $\xi_{N(t)} \geq \ell$ such that the interpolation grid (8) covers the whole interval $[0,\ell]$. This is the reason for our assumption that the system (1) is defined for all $x \geq 0$ instead of $x \in [0,\ell]$ (see Remark 1).

3 IMPLEMENTATION

The above Algorithm has been realized in Simulink in the form of a masked subsystem `FOQL PDE` ("first-order quasilinear PDE") which can be found in the library `PDECharactLib.slx` at (Ponomarev et al., 2020). An example Simulink model (Fig. 1a) which uses the subsystem can also be found there under the name `FOQL_PDE_SimpleExample.slx`. It implements (1) with $\ell = 1$, velocity $v = x + 1.1 + \sin t$, source/sink term $f = -w(t,x)$, boundary input
$$u = \begin{cases} \frac{1}{2}(1 + \cos\frac{\pi t}{2}), & \text{if } t \leq 16, \\ 0, & \text{otherwise,} \end{cases}$$


t
x
w
equationTerms
v
f
u
First-order quasilinear PDE:
w_t(t, x) + v(t, x) w_x(t, x) = f(t, x)
w(t, 0) = u(t)
x
w
i1
i2
FOQL PDE
output
y

(a) Simulink model of the dynamics (1). The algorithm of Section 2.2 is implemented in the `FOQL PDE` block.



(b) The inside structure of the `FOQL PDE` block.

Figure 1: Implementation of the method of characteristics in Simulink.

boundary output $y = w(t,1)$, and initial state function $w_0(x) = 1 - x^2$. The reader is advised to download and inspect the model as description below is kept brief. The *mask parameters* of the `FOQL PDE` block are:

- “Domain length (L)” – corresponds to ℓ.
- “Maximum number of characteristics (Nmax)” – hard upper bound on the number of characteristics $N(t)$ for the purpose of static memory allocation.
- “x-values” and “w-values” describe the initial function $w_0(x)$ by a grid of its values $(x, w_0(x))$.
- “Δx”, “Δw”, and “Δt” correspond to the parameters Δ_x, Δ_w, and Δ_t of the Algorithm.
- “Tolerance for crossing characteristics” – sets the allowable upper bound on $\xi_i(t) - \xi_{i+1}(t)$. Crossing characteristics (i.e., the event $\xi_i(t) \geq \xi_{i+1}(t)$) indicate a shock wave in the solution which makes the method of characteristics not physically sound (Polyanin et al., 2002). Simulation will be terminated in this case with an error message. Setting the tolerance to a small positive number, firstly, helps to avoid termination caused by numerical inaccuracies rather than actual shock wave and, secondly, allows jumps in the solution. The jump is when $\xi_i(t) \equiv \xi_{i+1}(t)$ but $\omega_i(t) \neq \omega_{i+1}(t)$. At zero tolerance, this situation would be interpreted as two characteristics crossing though in fact it may be a valid discontinuous solution (see also Remark 9).
- The check box “Terminate on overflow”, if checked, causes an error message when the current number of characteristics $N(t)$ equals its up-

per bound specified in the field “Nmax” above and the algorithm requires that another characteristic be created; otherwise, the simulation will silently continue without creating the characteristic. Unsetting the check box suppresses the characteristic creation trigger and may adversely affect the quality of the solution. The accuracy estimation given by the Theorem can no longer be guaranteed in this case.

The insides of the `FOQL PDE` block are exposed in Fig. 1b. The outputs `x` and `w` are constant-length vectors, their size specified in the mask parameter “Nmax”. They contain the variable-length vectors $\Xi(t)$ and $\Omega(t)$ from (5) as sub-vectors. To facilitate adding and removing elements to and from $\Xi(t)$ and $\Omega(t)$ according to the Algorithm, vectors `x` and `w` are endowed with *cyclic buffer* structure, meaning that the span of the elements containing $\Xi(t)$ and $\Omega(t)$ may wrap around the ends of `x` and `w`. The outputs `i1` and `i2` are the head and tail of the buffer, i.e., the indices at which the first characteristic (ξ_1, ω_1) and the last one $(\xi_{N(t)}, \omega_{N(t)})$ are stored in `x` and `w`.

Two topmost *Integrator* blocks in Fig. 1b are used to solve the characteristic equations (7), and the rest are employed to store `i1`, `i2`, and $t_{\mathrm{LC}}(t)$. Creation and removal of the characteristics is done by triggering the *Integrator* blocks’ state reset.

4 EXAMPLE

The repository (Ponomarev et al., 2020) contains three example models with the `FOQL PDE` block: `FOQL_PDE_SimpleExample.slx` is mentioned in the previous section, `FOQL_PDE_OutputFeedback.slx` is a real-life model of heat exchanger with output feedback control, and `FOQL_PDE_StateFeedback.slx` is an illustrative plant with full-state feedback. Here we discuss the latter and compare the solution produced by the `FOQL PDE` block to those obtained using the space discretization-based *Method of Lines* (MOL).

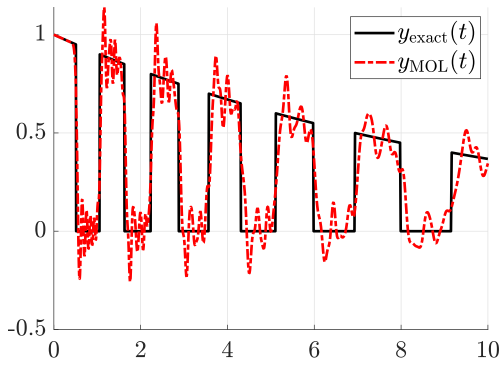
The plant under consideration is

$$\begin{aligned} w_t + v w_x &= \gamma w(t,x), \quad x \in [0,1], \\ w(0,x) &= w_0(x), \\ w(t,0) &= u(t), \\ \text{output: } y(t) &= w(t,1) \end{aligned} \tag{13}$$

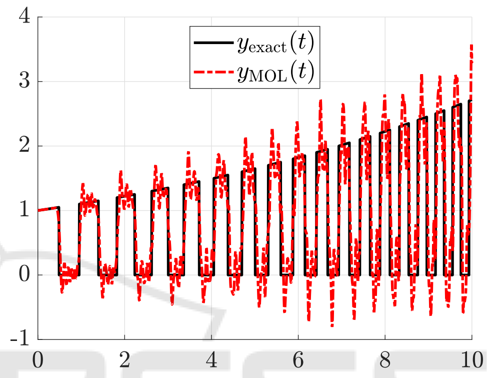
with a tunable constant γ and full-state feedback

$$v = v\big(w(t,\cdot)_{[0,1]}\big) = 2\int_0^1 w(t,x)\,\mathrm{d}x, \tag{14a}$$

$$u(t) = w(t,1). \tag{14b}$$



(a) $\gamma = -0.1$



(b) $\gamma = 0.1$

Figure 2: Output $y(t)$ of (13)-(15) obtained via the MOL with central finite differences and order $K = 100$ compared to the exact solution (16). The wiggles reveal numerical problems of the method.

Assuming the initial state is

$$w_0(x) = \begin{cases} 0, & \text{if } x < \frac{1}{2}, \\ 1, & \text{otherwise,} \end{cases} \tag{15}$$

one can obtain the exact output

$$y_{\mathrm{exact}}(t) = \begin{cases} a(t), & \text{if } s(t) - \lfloor s(t) \rfloor < 1/2, \\ 0, & \text{if } s(t) - \lfloor s(t) \rfloor \geq 1/2 \end{cases} \tag{16}$$

where $\lfloor \cdot \rfloor$ is the rounding-down operation and

$$a(t) = \mathrm{e}^{\gamma t}, \quad s(t) = \begin{cases} t, & \text{if } \gamma = 0, \\ \dfrac{a(t)-1}{\gamma}, & \text{otherwise.} \end{cases}$$

For the purposes of simulation, the integral in (14a) is approximated using the trapezoidal rule.

As the first approach to simulating the dynamics (13)-(15) we consider MOL with central finite difference approximations of the spatial derivative (Schiesser, 2012) with K discretization segments:

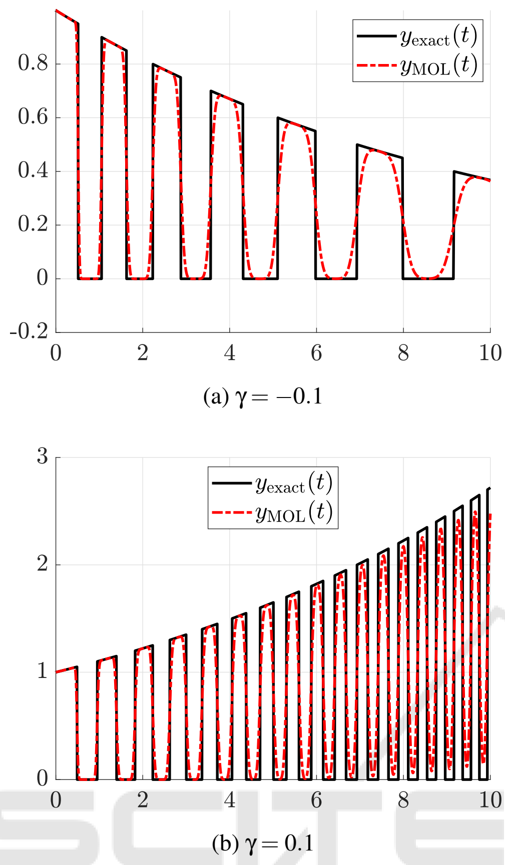


Figure 3: Output $y(t)$ of (13)-(15) obtained via the MOL with upwind scheme and order $K = 1000$ compared to the exact solution (16). Numerical diffusion destroys the shape of the solution.

$$\begin{aligned} w_{1,x}(t) &:= w_x(t,x_1) \approx \frac{w_1(t) - 2w(t,0) + w_2(t)}{2\Delta x}, \\ w_{i,x}(t) &:= w_x(t,x_i) \approx \frac{w_{i+1}(t) - w_{i-1}(t)}{2\Delta x}, \\ w_{K+1,x}(t) &:= w_x(t,x_{K+1}) \approx \frac{w_{K+1}(t) - w_K(t)}{\Delta x} \\ &\bigl(i = 2,3,\ldots,K\bigr) \end{aligned} \tag{17}$$

where $w_i(t) := w(t,x_i)$, $x_i := (i-1)\Delta x$ and $\Delta x := 1/K$. Applying (17) to (13) yields the ODE system

$$\begin{aligned} \dot{w}_i(t) &= -v\,w_{i,x}(t) + \gamma w_i(t), \\ w_i(0) &= w_0\left((i-1)\Delta x\right) \\ &\bigl(i = 1,2,\ldots,K+1\bigr). \end{aligned}$$

The results of this approach are shown in Fig. 2.

The wiggles in the MOL solution appear because the initial function w_0 from (16) is non-monotonic. Indeed, in general, finite difference-based MOL solutions suffer numerical diffusion, i.e., amplitude errors (smearing), and/or dispersion, i.e., phase errors (wiggles). In order to alleviate the effects, an upwind scheme for the spatial derivative, i.e.,

$$\begin{aligned} w_{1,x}(t) &:= w_x(t,x_1) \approx \frac{w_1(t) - w(t,0)}{\Delta x}, \\ w_{i,x}(t) &:= w_x(t,x_i) \approx \frac{w_i(t) - w_{i-1}(t)}{\Delta x} \\ &\bigl(i = 2,3,\ldots,K+1\bigr) \end{aligned}$$

has been implemented (Zijlema, 2015). The results are smooth (Fig. 3), however, a high resolution of discretization is required ($K = 1000$). Furthermore, the numerical solution departs from the exact one rather quickly due to numerical diffusion.

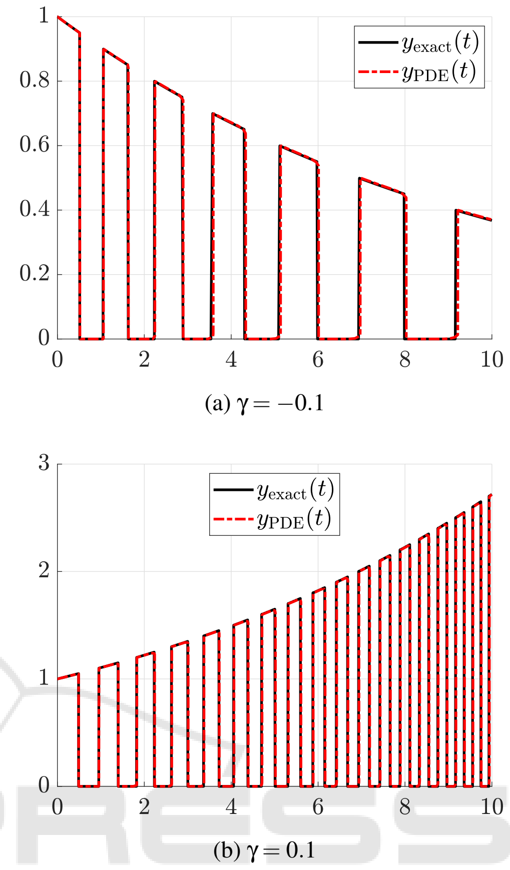


Figure 4: Output $y(t)$ of (13)-(15) obtained via the proposed `FOQL PDE` block compared to the exact solution (16). The disadvantages of the MOL are avoided to a large extent.

Finally, Fig. 4 presents the results of simulation using our characteristics-based `FOQL PDE` block. It yields better results than the MOL as neither numerical dispersion nor diffusion are apparent.

Remark 7. Further improvement of the MOL solution with upwind scheme could be achieved by creating higher order, monotone, nonlinear upwind schemes using, e.g., the flux limiting technique (Zijlema, 2015). However, flux limiters such as Minimod, Superbee, or Van Leerm limiter, are difficult to implement due to their nonlinearity.

Remark 8. Performance of the `FOQL PDE` block can be enhanced by adjusting the parameters of the block itself as well as the Simulink solver configuration. For instance, to obtain the results in Fig. 4 we had to set "Max step size" in Simulink's *Solver Configuration Parameters* to 0.1 and reduce the `FOQL PDE` block's parameter "Δw" to 0.01.

Remark 9. The discontinuous initial state (16) has been defined by setting the `FOQL PDE` block parameter "x-values" to `[0, 0.5, 0.5, 1]` and "w-values" to `[0, 0, 1, 1]`. Thus, two characteristics are initialized at the same point of the initial boundary with different "w-values", specifically, $\xi_2(0) = \xi_3(0) = 0.5$, $\omega_2(0) = 0$, $\omega_3(0) = 1$. This approach lets one define initial state with a jump exactly. It is allowed thanks to non-zero "Tolerance for crossing characteristics". At zero tolerance simulation would fail immediately.

To sum up, the finite difference-based MOL often yields inaccurate results due to the described numerical artifacts (diffusion and dispersion) whereas the results produced by the `FOQL PDE` block agree well with the exact solution.

5 CONCLUSIONS

The method of characteristics has been implemented in Simulink in the form of a masked subsystem which can be set up to simulate a wide range of dynamics described by first-order quasilinear PDEs. The solution showed good accuracy surpassing that of the discretization-based method of lines.

We conclude that the subsystem is a viable option for simulation of convection-reaction dynamics in Simulink. In the future, the approach may be extended to a larger class of equations such as first-order nonlinear PDEs and higher-order hyperbolic PDEs.

REFERENCES

Greenkorn, R. (2018). *Momentum, heat, and mass transfer fundamentals*. CRC Press.

Herrán-González, A., Cruz, J. M. D. L., Andrés-Toro, B. D., and Risco-Martín, J. L. (2009). Modeling and simulation of a gas distribution pipeline network. *Applied Mathematical Modelling*, 33(3):1584–1600.

Ingham, D. B. and Pop, I. (1998). *Transport phenomena in porous media*. Elsevier.

Kachroo, P. and Özbay, K. M. (2018). Traffic flow theory. In *Feedback Control Theory for Dynamic Traffic Assignment*, pages 57–87. Springer.

Mahmoudi, Y., Hooman, K., and Vafai, K. (2019). *Convective heat transfer in porous media*. CRC Press.

Polyanin, A. D., Zaitsev, V. F., and Moussiaux, A. (2002). *Handbook of first order partial differential equations*. Taylor & Francis.

Ponomarev, A., Hofmann, J., and Gröll, L. (2020). PDECharactSimulink. https://github.com/antonponmath/PDECharactSimulink. [Online].

Schiesser, W. E. (2012). *The numerical method of lines: Integration of partial differential equations*. Elsevier.

Truskey, G. A., Yuan, F., and Katz, D. F. (2004). *Transport phenomena in biological systems*. Pearson/Prentice Hall.

Van Schijndel, A. (2009). Integrated modeling of dynamic heat, air and moisture processes in buildings and systems using Simulink and COMSOL. *Building Simulation*, 2:143–155.

von Petersdorff, T. (2012). MATLAB files `quasilin.m` and `colorcurves.m`. http://www2.math.umd.edu/~petersd/462/. [Online].

Witrant, E. and Niculescu, S.-I. (2010). Modeling and control of large convective flows with time-delays. *Mathematics in Engineering, Science and Aerospace*, 1(2):191–205.

Zhang, F. and Yeddanapudi, M. (2012). Modeling and simulation of time-varying delays. In *Proceedings of the 2012 Symposium on Theory of Modeling and Simulation-DEVS Integrative M&S Symposium*, pages 34–41. Society for Computer Simulation International.

Zijlema, M. (2015). *Computational modelling of flow and transport*. Faculty of Civil Engineering and Geosciences, Delft University of Technology.